(20)

A TRIDENT SCHOLAR PROJECT REPORT

NO. 205

"APPLICATIONS OF NEURAL NETWORKS IN FAULT DETECTION OF ROTATING MACHINERY"

UNITED STATES NAVAL ACADEMY
ANNAPOLIS, MARYLAND

U.S.N.A. Trident Scholar project report; no.205 (1993)

"APPLICATIONS OF NEURAL NETWORKS IN FAULT DETECTION OF ROTATING MACHINERY"

by
Midshipman William O. Nash Jr, Class of 1993
U.S. Naval Academy
Annapolis, Maryland

[illegible]
Advisor: Assistant Professor William I. Clement
Weapons & Systems Engineering Department.

Accepted for Trident Scholar Committee

Francis D Correll
Chair

May 17, 1993
Date

USNA-1531-2

REPORT DOCUMENTATION PAGE	*Form Approved* *OMB no. 0704-0188*

1. AGENCY USE ONLY (Leave blank)	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	May 17, 1993	

4. TITLE AND SUBTITLE

Applications of neural networks in fault detection of rotating machinery

6. AUTHOR(S)

William O'Neal Nash, Jr.

7. PERFORMING ORGANIZATIONS NAME(S) AND ADDRESS(ES)	8. PERFORMING ORGANIZATION REPORT NUMBER
U.S. Naval Academy, Annapolis, MD	U.S.N.A. - Trident scholar project report ; no. 205

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)	10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

Accepted by the U.S. Trident Scholar Committee

12a. DISTRIBUTION/AVAILABILITY STATEMENT	12b. DISTRIBUTION CODE
This document has been approved for public release; its distribution is UNLIMITED.	

13. ABSTRACT (Maximum 200 words)

The purpose of this research was to design a neural network based fault diagnosis system that is capable of detecting and classifying incipient faults in rotating machinery.

A neural network is essentially a pattern recognition system which produces a mapping from a set of input data to a set of output data. Neural networks are unique in that this mapping is created autonomously, based on a learning algorithm that the user specifies. In this research, the mapping is from a set of measured parameters (e.g., vibration spectrum) of the rotating machinery to a classification of the systems's condition (e.g., worn bearings).

Limited success has been achieved in this area over the past decade. Research to date indicates that neural networks have the ability to recognize faults in machinery. This research focused on the following objectives: (1) creating a system that can recognize basic fault conditions based on the fault's vibration signature; (2) improving the ability of the neural network to recognize transient conditions as normal rather than classify them as faults; (3) examining the effect that disturbances have on the neural network's output; and (4) developing a system that will enable detection of faults which are not included in the training set.

14. SUBJECT TERMS	15. NUMBER OF PAGES
Neual networks, fault detection, transients, backpropagation	49
	16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATATION OF ABSTRACT
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED

PREFACE

I would like to take this opportunity to thank my Trident Advisor, Assistant Professor William Clement for the guidance that he provided throughout the past year. Without his help, this research could not have been completed. A special thanks also goes out to everyone in the Technical Support Division for their help. Finally, I would like to thank my parents, Anthony and Deborah Dupree for the support that they have given me.

TABLE OF CONTENTS

INTRODUCTION

The focus of this research was the development of a neural network based condition monitoring system. The purpose of such a system is to prolong machinery life by ensuring that it operates within safe parameters. The goal is to allow preventive maintenance to be performed in such a manner as to prevent catastrophic failures and to limit down time.

Currently, there are two categories of maintenance that are performed on equipment. The first form is corrective maintenance. Corrective maintenance is performed when a machine is operated to the point of breakdown, at which time technicians "correct" the problem and restart the machine. The disadvantages of this approach are that there is often a lot of down time and the machine may suffer a substantial amount of damage, making repair costly.

The second form of maintenance is preventive maintenance. Preventive maintenance is an integral part of the military's daily routine. In the military, corrective maintenance is an unacceptable alternative. Equipment that does not function risks lives and reduces the effectiveness of a unit's assets.

However, current methods of preventive maintenance are based on periodic schedules. These schedules were established based on empirical data that indicate when a machine needs to be serviced. They do not reflect actual knowledge of any particular machine's operating conditions. The result is that machines may be serviced when they do not need to be and may not get serviced when they do. Neither case is economically efficient.

The ideal situation would be to incorporate a condition monitoring system which allows preventive maintenance to be scheduled more efficiently. Current methods of condition monitoring are based on analysis by skilled technicians. The technicians examine data which reflect the machine's operating parameters and then make a judgement on whether or not the machine is operating properly.

This approach has several limitations. First, the experience of the technician has a direct relationship on his/her ability to distinguish parameters which might lead to a fault condition. Not all technicians have the same level of expertise. Second, the training and use of these technicians is expensive. Finally, consistent performance is difficult to guarantee. People perform differently under different situations. A technician who has been on watch for six hours may not recognize something that he would have recognized had he just posted the watch.

The use of artificial neural networks may provide an answer to some of these problems. An artificial neural network creates a vector mapping between two sets of data. In this case, the input is a vector of machine descriptors and the output is the machine's current condition. The advantage of artificial neural networks is that they do not require a technician, skilled in data processing, to create the mapping. This reduces cost by taking people out of the loop and it offers a much more consistent level of performance since it is not subject to human factors such as boredom, fatigue, and incompetence.

HISTORY OF NEURAL NETWORKS

The development of artificial neural networks was the result of man's attempt to simulate the operation of the human brain. The brain consists of billions of densely packed, interconnected neurons. The connections between neurons are called synapses. The transmission of information across these synapses is electro-chemical in nature, with the strength of the signal depending on the amount of neuro-transmitter released. These synapses, and the information that they transfer, form the basis for the operation of the human brain.

The construction of an artificial neural network is very similar to the model of the brain. It consists a large number of processing elements that are connected in a highly parallel manner. The strengths of the various connections are determined by variable weighting factors. The power of neural networks lies not in the complexity of individual neurons but, rather, in the high degree of parallelism which exists.

Research on neural networks has been an ongoing process for the last forty five years. Interest in this field was sparked in the late 1940's when McCulloch and Pitts published a paper entitled, "A Logical Calculus of Ideas Imminent in Nervous Activity". [Ref #1] Further research, in particular the efforts of Hebb and Rosenblatt, led to the development of what became known as the Hebbian rule [Ref #2] and the perceptron [Ref #3]. Later, Widrow and Hoff developed a neural network based on the ADALINE (ADAptive LINear Element) perceptron [Ref #4]. They correctly theorized that it would be possible for their network to distinguish between two linearly-separable regions. Although this theory was valid, their work was criticized harshly in the book, Perceptrons, written by Marvin Minski and Seymour Pappert [Ref #5]. After extensive mathematical study, Minski and Pappert concluded that a neural network based on the ADALINE perceptron was unable to solve non-linear problems such as the exclusive-OR problem. They also found that the network did not have the ability to adjust the weights of hidden layers. This severely limited the amount of learning that the neural network could achieve. Although some research did continue in the field, this book's conclusion that neural networks did not warrant further study, caused both interest and funding for study to be drastically reduced.

In the early 1980's, neural networks again began to attract the attention of scientists. In 1986, Rummelhart refined a learning process that had become known as the generalized delta rule [Ref #6], which was a means of adjusting the weights within a neural network. This allowed learning algorithms to adjust the hidden layer weights. Since his findings, interest in neural networks has steadily increased. Practical applications are continually being discovered. Currently, neural networks are being used for such applications as stock market forecasting, signal processing, check reading and verification. AT&T uses a neural network chip to reduce the noise on phone lines.

The advantage that neural networks offer is that an operator does not have to have an a priori knowledge of the equations which relate the input data to the output data. This provides a powerful tool for analyzing data for slight discrepancies that are not immediately obvious.

WHAT IS A NEURAL NETWORK?

Although artificial neural networks are based on a model of the human brain, they do not learn in the same manner that humans learn. The operation of neural networks is statistical in nature. This was proven in 1989 when University of California San Diego economist Halbert White reduced the backpropagation procedure to a stochastic approximation. [Ref #7]

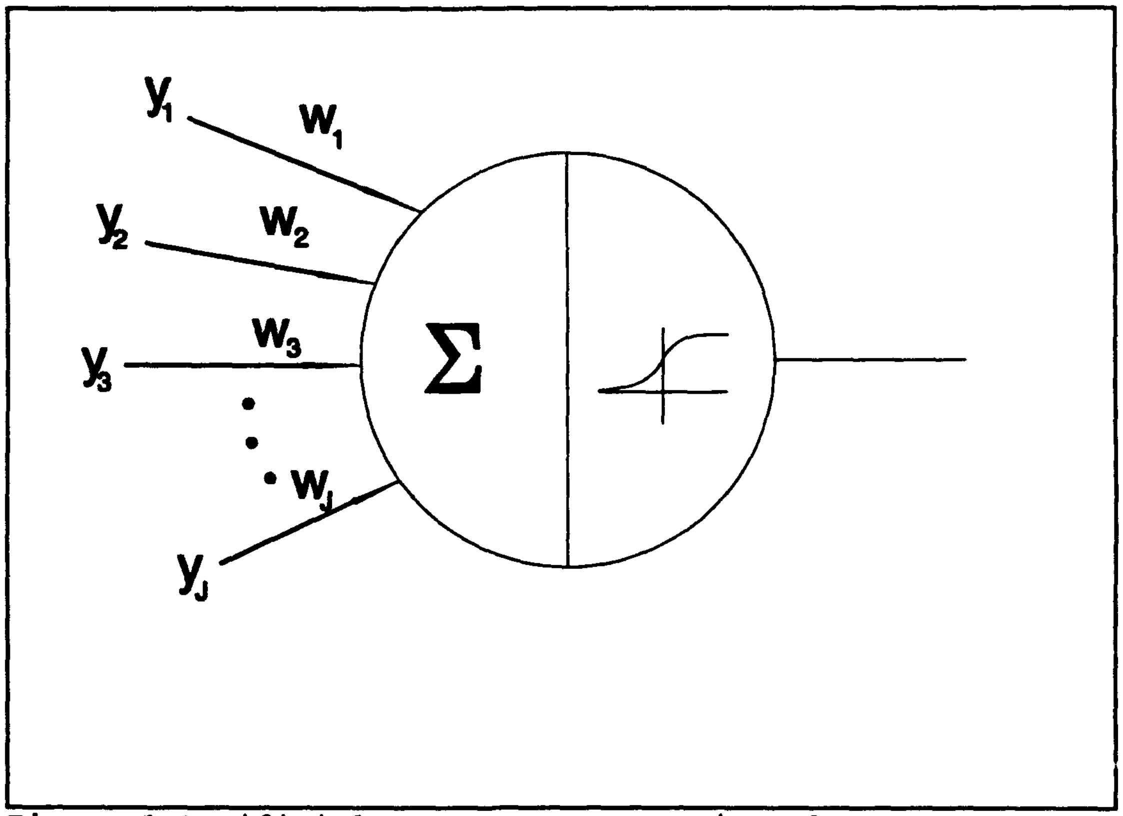


Figure 1 Artificial neuron or processing element

The basic element of the artificial neural network is the artificial neuron or processing element (Fig 1). Each processing element receives input from many other neurons. Each input is multiplied by a synaptic weight and these weighted inputs are then summed and passed throu$_{\mathit{f}}$'. an activation function to form the neuron's output. This process is repeated as the output of the neuron is fed into other neurons.

The activation function that is used within the individual neuron is a continuous function. Typical functions that are used for this purpose are the sigmoid function, the sine function, and the hyperbolic tangent function. These functions are used because of the ability to express their derivatives in terms of the original function facilitates programming. The equation for the sigmoid function is

$$f(net_k) = \frac{1}{1+e^{-net_k}} \tag{1}$$

The typical artificial neural network has its neurons grouped into layers. The input layer performs no processing but merely buffers the input data and passes it on to succeeding layers. The final layer is where network output is read. Between these two layers may be any number of hidden layers. The purpose of each hidden layer is to extract information from the previous layer's outputs for mapping to the next layer. In general, more hidden layers equates to more complex mappings.

Two characteristics are needed to accurately describe a neural network. The first is its architecture and the second is the type of learning algorithm that the neural network employs. It is a common misconception to label neural networks using only one of these specifications. A neural network cannot be accurately labeled by only one of these characteristics.

There are two basic types of neural network architectures. The first and most widely used is the feed- forward network. In the feed-forward network, the output of each processing element is fed forward to a neuron on a successive layer (Fig 2).

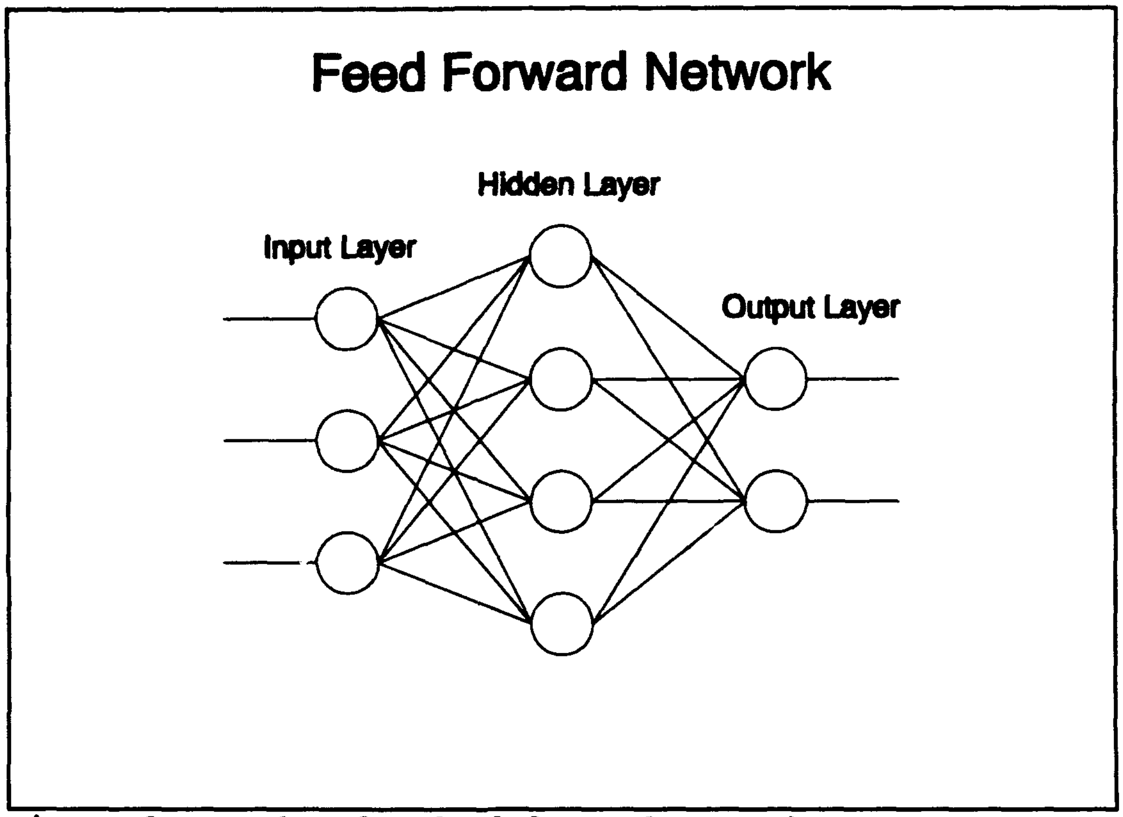


Figure 2 Example of a feed-forward network

The second type is the recurrent or feedback neural network. This neural network makes use of a feedback path from the output of one or more processing elements to the input of the processing elements of a previous layer. The advantage of a feedback network is that, like sequential circuits, the network does not always respond in the same to the same set of input data. It exhibits memory (state). The network's output depends both on its inputs and on its current state.

Neural network learning rules are also classified with one of two ways.

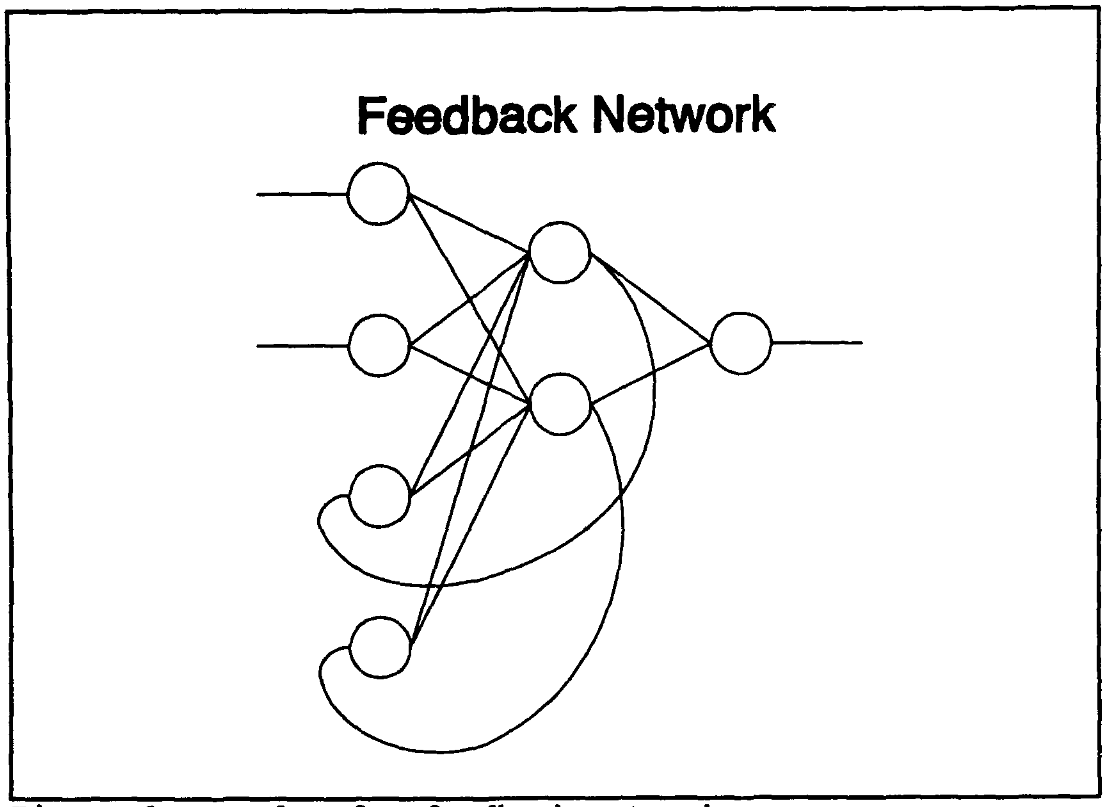


Figure 3 Example of a feedback network

Supervised learning takes place when both the output and input data are known (for some training set) and are presented to the neural network. The neural network continues to adjust its synaptic weights until the network output converges to the desired output. At this point, the system has "learned" the training set of data and is ready for use.

The second category of learning is unsupervised learning. It is most applicable when the number of groupings of the input data is unknown. Unsupervised learning does not require that a desired output data be presented to the neural network. It analyzes the data and groups the input patterns based on characteristics that it determines within the feature input space. The level of sensitivity of these networks can be altered, which determines the number of categories into which the data are separated.

A key advantage of neural networks is that all of the information that the network uses to compute its output is processed at one time. The data are processed in parallel vice series which reduces the response time of the network. Thus, neural networks can be used in real time applications that require continuous monitoring.

BACKPROPAGATION

For this research, a feed-forward neural network with backpropagation (supervised) learning was deemed appropriate because of its success in the past. The name backpropagation was derived from the manner in which the changes in the synaptic weights are made. This algorithm seeks to adjust the weights in such a manner as to follow the steepest gradient descent of the RMS error function in order to reach a least mean square error between the actual and desired output of the network.

The backpropagation algorithm assumes that all processing elements are responsible for the output error. As a result, output error is propagated backwards (from output to input layer), in order to assign blame to each weight in proportion to its effect. The backpropagation routine operates in the following manner. First, the weights of the neural network are initialized to small random values near zero. If this is not performed, then the network will not train since it will be performing calculations with matrices of zeroes. Second the training vector **y** is presented to the neural network and the output vector **o** is calculated, where

$$\mathbf{y}=\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_J \end{bmatrix}, \mathbf{o}=\begin{bmatrix} o_1 \\ o_2 \\ \vdots \\ o_K \end{bmatrix} \tag{2}$$

The weight matrix is

$$W=\begin{bmatrix} w_{11} & w_{12} & \cdots & w_{1J} \\ w_{21} & w_{22} & \cdots & w_{23} \\ \vdots & \vdots & \cdots & \vdots \\ w_{K1} & w_{K2} & \cdots & w_{K3} \end{bmatrix} \tag{3}$$

The output matrix is calculated by

$$\boldsymbol{o}=\Gamma[\boldsymbol{W}\boldsymbol{y}] \tag{4}$$

where

$$\Gamma[\cdot]=\begin{bmatrix} f(\cdot) & 0 & \cdots & 0 \\ 0 & f(\cdot) & \cdots & 0 \\ \vdots & \vdots & \cdots & \vdots \\ 0 & 0 & \cdots & f(\cdot) \end{bmatrix} \tag{5}$$

The desired output of the neural network is

$$\boldsymbol{d}\triangleq\begin{bmatrix} d_1 \\ d_2 \\ \vdots \\ d_K \end{bmatrix} \tag{6}$$

The RMS error of the system is then calculated by taking the square of the difference between the desired output and actual output data.

$$E_p=\frac{1}{2}\sum_{k=1}^{K}\sqrt{(d_{pk}-o_{pk})^2}=\frac{1}{2}\sqrt{|d_p-o_p|^2} \tag{7}$$

where the subscripts p and k refer to the training pattern and the network layer, respectively. Next, the error associated with each neuron is calculated and the weights are adjusted by calculating the error, as a function of the weight in question. Training continues until the RMS convergence criterion is met. The goal of this procedure is to alter the weights of the processing elements in order to minimize the RMS error. Because of the continuous nature of the activation function (sigmoid), the RMS error never reaches zero.

This process can be visualized by examining a marble that is placed on top of a large mountain range that contains many local minima (valleys). The goal is to roll the marble ι ·wn the mountain range so that it meets two criteria. First, it is desirable for the marble to follow the steepest path in order to allow it to get to its destination quickly. Second, the marble should come to rest at the lowest point on the mountain range in order to minimize the RMS error, which is symbolized by the marble's altitude.

The rule which governs the marble's descent down the mountain range is the delta rule. It determines the size of weight adjustment within the neural network. It is given by:

$$\Delta w_{kj}=\eta\delta_k y_j \tag{8}$$

where the subscripts j and k refer to the node and layer, respectively. The constant η is the learning coefficient, δ_k is the error signal as a function of the weights and y is the input vector. At this point, the delta rule can be derived. Each individual weight adjustment is given by

$$\Delta W_{kj}=-\eta\frac{\partial E}{\partial w_{kj}} \tag{9}$$

where the error is determined as in Eqn (2). For each node in layer k, k=1,2,...K the following holds:

$$net_k=\sum_{j=1}^{J} w_{kj}y_j \tag{10}$$

where the subscript j refers to each node within each layer. Each neuron's output is

$$o_k=f(net_k) \tag{11}$$

The error signal term δ_{ok} provided by the kth neuron is defined for this layer as

$$\delta_{ok} \triangleq -\frac{\partial E}{\partial (net_k)} \tag{12}$$

Since the gradient component depends only on the net_k of a single neuron, and since the output of the kth neuron is contributed to only by weights w_{kj}, j=1,2,..J for a fixed K, the chain rule allows us to write

$$\frac{\partial E}{\partial w_{KJ}} = \frac{\partial E}{\partial (net_k)} \cdot \frac{\partial (net_k)}{\partial w_{kj}} \tag{13}$$

Since the values of y_j for j=1, 2, ... J, correspond to a fixed pattern at the input,

$$\frac{\partial (net_k)}{\partial w_{kj}} = y_j \tag{14}$$

Combining Eqn(12) and Eqn(15) yields

$$\frac{\partial E}{\partial w_{kj}} = -\delta_{ok} y_j \tag{15}$$

Therefore

$$\Delta w_{kj} = \eta \delta_{ok} y_j, k=1,2,...K \tag{16}$$

Since E is a function of net_k

$$E(net_k)=E[o_k(net_k)] \tag{17}$$

and from Eqn(12)

$$\delta_{ok}=-\frac{\partial E}{\partial o_k}\cdot\frac{\partial o_k}{\partial(net_k)} \tag{18}$$

The second term in the product of eqn (16) is the derivative of the activation function

$$f'(net_k)\triangleq\frac{\partial o_k}{\partial(net_k)} \tag{19}$$

and

$$\frac{\partial E}{\partial o_k}=-(d_k-o_k) \tag{20}$$

Thus

$$\delta_{ok}=(d_k-o_k)f_k'(net_k), \tag{21}$$

which gives the delta rule

$$\Delta w_{kj}=\eta(d_k-o_k)f_k'(net_k)y_j \tag{22}$$

From this result, the generalized delta rule can derived. This derivation will not be performed, however the equation governing the generalized delta rule is given by

$$\Delta v_{ji}=\eta f_j'(net_j)z_i\sum_{k=1}^{K}\delta_{ok}w_{kj}, \tag{23}$$

where v is the weight matrix of the connections to the hidden layer, and z_i is the input to the hidden layer. w denotes the synaptic weights of the layer following the hidden layer.

The generalized delta rule can be modified so that it includes a momentum term which acts as a low pass filter, since general trends within the weight space are reinforced

$$\Delta w_{kj}=-\eta\nabla E(t)+\beta\Delta w(t-1) \tag{24}$$

while high frequency variations are ignored. This allows smaller learning coefficients to be used which in turn let the neural network examine the weight space in greater detail. In Eqn (24) the momentum term is given by ß, where t and (t-1) indicate the current and most recent training step. Using the marble analogy, the addition of the momentum term helps to prevent the marble from becoming stuck in a shallow valley rather than the deep crevice for which it is searching.

The advantages of using a multi-layer, feed-forward backpropagation network are that it is relatively easy to implement and has powerful classification abilities. The most noteworthy of these abilities is the classification of non-linearly separable pattern classes. This allows neural networks to be used to solve a wide variety of problems such as the exclusive-or problem

VIBRATION MONITORING

Vibration monitoring "consists of acquiring vibration signals and using them as information carriers." [Ref #8] This process is suitable for this project for the following reasons: (1) vibration monitoring provides a non-intrusive means of examining the system; and (2) reliable transducers, which can sense these vibrations, are readily available. Of course, the most important reason is that the vibration spectrum is strongly affected by the machinery operating condition.

Several analysis methods are used in traditional vibration monitoring systems. The most widely used is frequency domain analysis. Since we must sample the vibration data, the conversion from time domain to frequency domain is accomplished by taking a discrete Fourier Transform (DFT) of the sequence. The DFT X(k) of a sampled time waveform x(t) is given by:

$$X(k)=\sum_{k=0}^{N} x(k)e^{-(j2\pi kn/N)}, n=0,1,\ldots N-1$$

Computation time for the FFT is of order $O(n^2)$ where n is the number of elements in the sequence. For large sequences, this becomes burdensome. However, if n is a power of 2, a Fast Fourier Transform (FFT) may be employed. This reduces computation time to the order $O(n\log(n))$.

Analysis in the frequency domain provides useful information when the time domain data describe oscillating phenomena such as those that are encountered in rotating machinery. Many of the faults that occur in these types of machinery result in perturbations that occur at specific frequencies (Table I). Ideally, the neural network will be able to identify these frequencies and relate them to their respective fault conditions.

Table I: Typical rotating machinery faults and the frequencies that these faults produce.

Source	Frequency
Imbalance	f_r (frequency of shaft rotation)
Bent Shaft	f_r,$2f_r$,and $4f_r$
Mechanical Looseness	$2f_r$
Faulty Belt Drive	multiple of belt frequency
Misaligned Race	$f_c \pm 2f_r$

Several other methods are also of interest when dealing with rotating machinery. One such method is the cepstrum. The cepstrum is defined as the inverse fourier transform of the logarithm of the power spectrum. Cepstrum analysis has been praised because of its ability to filter out the effects that the structure of the device may have on the measured signal. W.D. Strunk found cepstrum analysis to be especially helpful in identifying bearing problems in pieces of rotating machinery. [Ref #9]

Structural masking is a phenomenon that occurs when the vibration signal is modified by the dynamic properties of the structure. In some cases, this modification can be severe, resulting in data that are representative of the structure rather than the machinery's vibration signal [Ref #8]. This is desirable for some applications. If the integrity of the structure is a concern, then data which are representative of the structure are useful. In this study however, care should be taken to ensure that the data represent the machinery.

The focus of this research did not require the use of powerful feature extraction techniques. The spectrum alone provided sufficient information for classification. However, vibration analysis serves a valuable purpose. The ability to examine the vibration data of a piece of machinery allows humans to study the condition of the machine. The data in Table I were obtained from vibration analysis of machine. By using vibration analysis, the amount of information that is presented to the neural network can be reduced drastically. This would make the neural network more efficient. The disadvantage is that this vibration analysis is time consuming. In this project, an attempt was made to limit the amount of resources that were devoted to vibration analysis methods.

DESIGN OF THE PHYSICAL SYSTEM

The physical system that serves as the model for our rotating machine is a simple device designed around a small AC motor. The AC motor is a 1/3 horsepower Dayton Wattrimmer, model number 3M384. The motor, or main shaft turns at 1075 rpms. It is coupled to a secondary shaft via a timing belt.

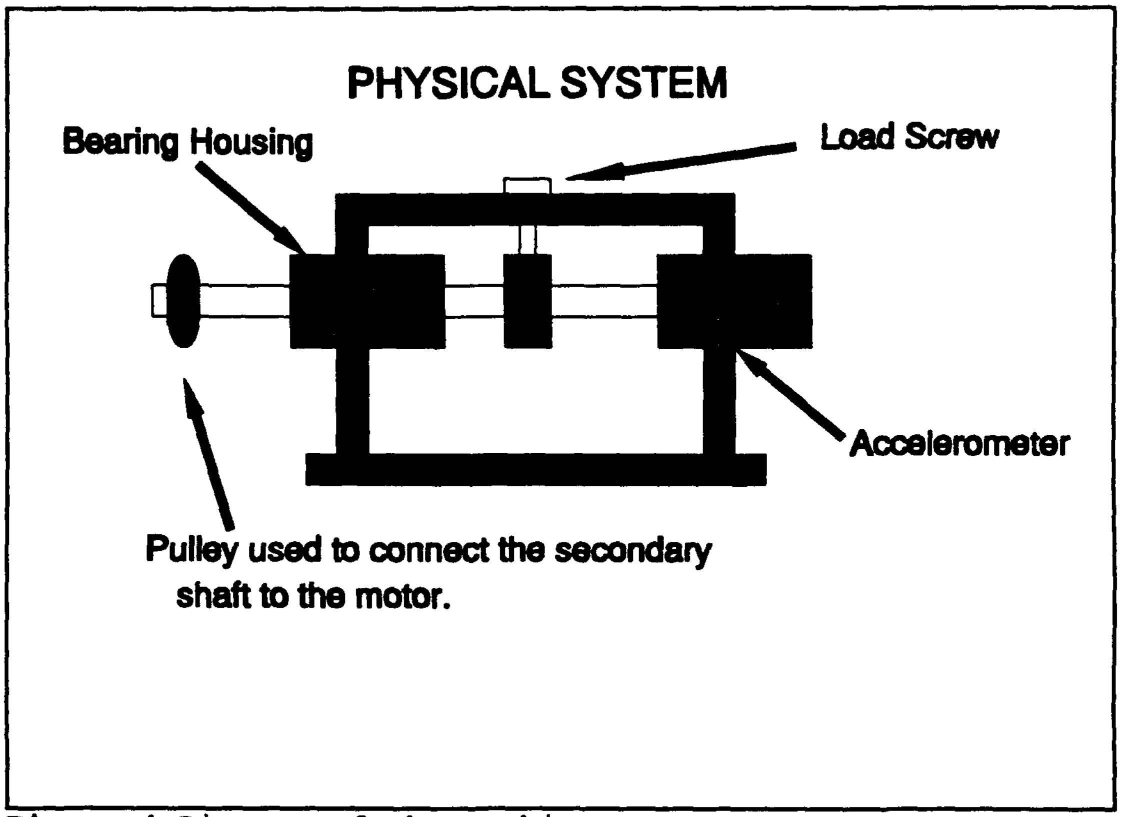


Figure 4 Diagram of the machine

All of the testing was performed on the secondary shaft. This shaft is attached to the framework of an aluminum housing that serves to support the shaft (see Figure 4). The shaft passes through two sets of radial bearings which enable the shaft to rotate. In addition, it is possible to add a load to the shaft by tightening a bolt that is on top of the aluminum housing.

Each bearing on the secondary shaft is surrounded by a housing. These housings can be filled with a fluid or substance to accelerate bearing wear. A drain is provided so that the current bearing lubricant can be drawn off by rinsing the bearings with a solvent.

The vibration data that are collected from this work are gathered from an accelerometer, which was mounted radially above and on the side of the aluminum housing. The accelerometer was mounted via a 10-32 bolt that protrudes from the aluminum housing. The location of the accelerometer used for the project can be viewed in Figure 4.

Several faults can be induced in the system. Bearing wear is produced by the addition of a corrosive element to the bearing lubricant. Overloading the shaft can be simulated by the aforementioned technique. A fault can also be induced by altering the tension in the timing belt. Loosening the belt will cause it to slip, resulting in a new fault condition. The belt could also be worn so that the integrity of the interface between the belt and the gear is damaged. This could cause the belt to slip.

DESIGN OF THE MONITORING SYSTEM

The monitoring system consisted of the accelerometer which was mounted on the frame of the machinery. The signal from the accelerometer was amplified and then passed through an A/D converter. A software package then computed an FFT of this time domain data.

The amplifier that was constructed was built using a high input impedance op-amp, the LH0084. The schematic for this amplifier is included in Appendix I. It was chosen for its gain and precision. The peak to peak amplitude of the signal from the transducers is approximately 150 mV. Since the accelerometer was uncalibrated, it is unknown how many g's this corresponds to. This is unimportant, however, since the interest is in spectral lines and not absolute amplitudes. A gain of 100 (amplified so that magnitude is between -7.5 V and 7.5 V) was used so that the input signal covered a large portion -10V to +10 V range of the A/D converter. This increased the resolution of the system.

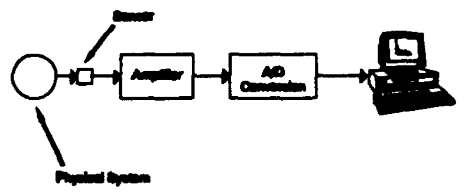


Figure 5 Flow chart of condition monitoring system

Some have advocated the use of a high pass filter in conjunction with the previously mentioned components. The high pass filter eliminates frequencies below a certain threshold frequency. The vibrations that are typically associated with bearing vibrations are located in the higher frequencies. It is was decided to not use a filter because of the neural network's ability to extract the features.

The A/D converter used a part of a Data Translation DT2801a data acquisition card. A program was written to control this device as well as to compute the FFT of the time domain data thus collected. This information is stored in a file and used to train the neural network. Software to perform these functions was available in the C programming language. This program is included in Appendix II. It further allows viewing of the data in either the time domain or frequency domain (with or without a Hanning window).

A Hewlett Packard HP3562A dynamic signal analyzer and an oscilloscope were also available for this study. These devices were used to examine various aspects of the system quickly, so as to select appropriate sampling characteristics for the A/D converter.

The heart of the neural network based monitoring system is the software package, Neural Works Professional II/Plus. This package was used to construct and test all of the neural networks created in this research. It is highly versatile in that it supports many different learning algorithms and network architectures.

RESULTS

The results of this research were obtained from a series of experiments that were designed to fulfill the stated objectives. These included a study of disturbances, transient conditions, and various fault conditions. The belt faults mentioned earlier in the report were not used due to time constraints.

Transients

The transient that was examined in this project was the vibrational signature of the machine. This signature varies during the period between warmup from a cold start to the point at which it reaches a steady state. Essentially, this problem is similar to the problem that a person has with an old automobile on a cold day. The machinery in this experiment, like an automobile, took a while to warm up and its performance was erratic until it did so. The purpose of this portion of the project was to examine the spectral data taken on the machine after a cold start in order to determine if the network could be trained to treat this transient condition as a normal operating condition.

Data for this first study were collected in five second segments at one minute intervals for thirteen minutes. In each five second interval, thirty 256 point FFTs were computed. Only the positive frequency components of the signal were retained due to the redundancy present. The resulting 128 point FFT spectrum was presented to the neural network as input. The frequency range that was used extended from 0 to 5 KHz. The neural network that was trained consisted of a 128 node input layer, a 1 node hidden layer, and a 1 node output layer (128-1-1). The activation function used was a sigmoid function and the delta rule was used to train the network. The desired network output for each piece of training data is listed in Table II. The values for desired output are values that were chosen to represent specific known conditions of the rotating machine.

Table II: Training criteria used to classify transients

Time (Min)	Desired Output
1	.9
3	.7
5	.7
7	.5
9	.3
11	.3
13	.1

The goal was to prove that the neural network could (1) train to recognize specific transient conditions and (2) make classifications on sets of transient data that it had never seen before. The theory to be tested in this experiment was that the changes in the machinery's frequency spectrum vary smoothly during this startup period.

The results of this experiment indicated that it was possible for the neural network to make a determination of the length of time after startup that a set data was taken. This can be seen in Figures 6 and 7. The bar chart in Figure 6 shows the response of the neural network to the set of data on which it was trained. This is not surprising in itself because, the network's response to the training data gain be estimated by looking at the RMS error. The information that is of interest is how the neural network responds to the data taken at times on which it was not trained. Figure 7 shows this graph and it is from this graph that certain conclusions can be drawn.

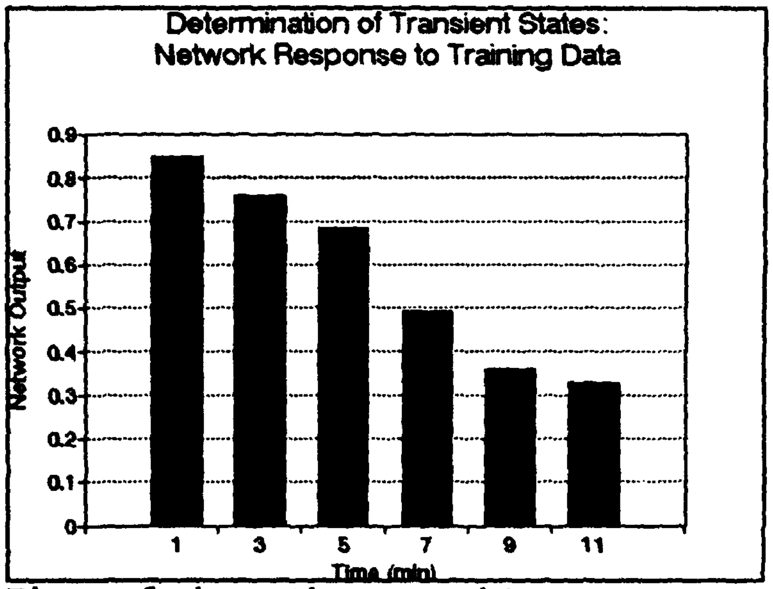


Figure 6 shows the network's response to the training vector. Results should be similar to values in Table II.

The bar chart in Figure 7 supports the conclusion that the frequency spectrum vary smoothly. It also supports the conclusion that a neural network can be trained to recognize transient conditions in order to prevent it from misclassifying them as fault conditions.

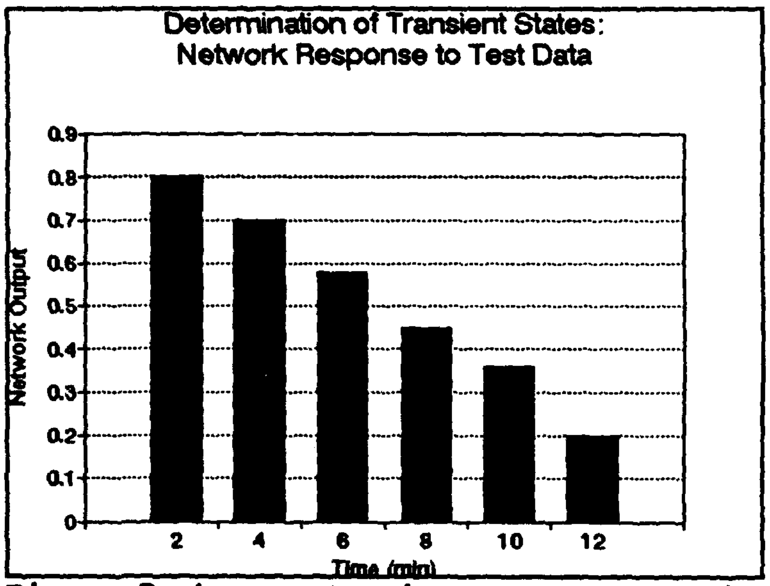


Figure 7 shows network's response to the test vector. The similarity of these results to the results in Figure 6 proves the validity of this method.

This test was not exhaustive, however. There may be other transient conditions on which the neural network was trained which would look like fault conditions. Testing must be undertaken on each transient to ensure that the network's classification is accurate.

Loading Effects

As discussed in the construction details of this paper, the physical system possessed a means of varying the radial loading placed on the secondary shaft. This was accomplished by tightening and loosening the load screw on top of the machinery. The purpose of this experiment was examine the ability of the neural network to identify different loading conditions applied to the secondary shaft. The data used to train the network were again in the form of a 128 point spectrum.

For this experiment, two sets of data were taken. The first was the control set. These data were taken with no loading placed on the shaft. The second set of data were taken after the load screw had been tightened to a hand tight position. These data served as an example of the fault condition. The data were taken in approximately 15 second segments (around 90 to a 100 FFTs were taken during this time) and were used to train a network. The network used in this experiment was again a 128-1-1 network.

The network was successfully trained on these data. However, when the network was presented a set of data upon which it had not been trained, it failed to make an accurate classification. These data were collected under identical conditions as the training data, but they were taken at different times. The reason for this failure was that the information that was presented to the neural network was not a statistically valid set of data. If the information presented to a neural network does not contain the information necessary to identify the various features of the training sets then the neural network cannot accurately classify the pattern. This is often referred to as the "garbage in, garbage out" theory.

A question that might be asked is, "How did the neural network's RMS error converge if the neural network was not presented a valid set of data upon which to train?" The answer to this question is that the neural network's classifications were erroneously based on differences within pattern space which resulted from disturbance noise. The descriptors chosen to train the network were not sufficient to separate the pattern classes. The successful convergence of the neural network during training indicates that the convergence was based on spurious patterns contained within the descriptors that did not contain information on the nature of a fault.

Another possible reason for the failure to classify the training data was the possibility of changing vibration characteristics of the machine. As a result, another test was devised. This test involved taking seven sets of data for both of the loaded and normal conditions of the machine and training the network on this information. Again, the network successfully trained on these data but failed to accurately classify the test data. This supported the earlier conclusion that the data presented to the network were not statistically valid.

Fault Conditions

The most important aspect of this project is the neural network's ability to detect and classify fault conditions. If the network cannot accurately classify and detect these faults, then the ability to handle transients and disturbances is wasted. The vibrations that are being studied are the bearing vibrations of the piece of machinery under three conditions of bearing wear. The conditions are listed in the following table along with their appropriate desired output data.

Table III: Desired response of the output nodes to various fault conditions.

Condition	Output Node 1	Output Node 2
Normal	0	0
Damaged	0	1
Non-lubricated	1	1

The normal bearing condition consisted of a bearing that was properly lubricated and free from any defects. The damage induced on the bearing in the next case was achieved by using a screwdriver to deform the housing near one of the individual ball bearings. Figure 7 shows a diagram of the bearing used and the location of its damage. The final condition was a bearing stripped of its lubricant by soaking it in a slightly acidic solution overnight.

The neural network used to examine this particular problem consisted of a 128-node input layer, a 2-node hidden layer, and a 2-node output layer. The file used to train the network was created by appending three separate files, each of which contained a five

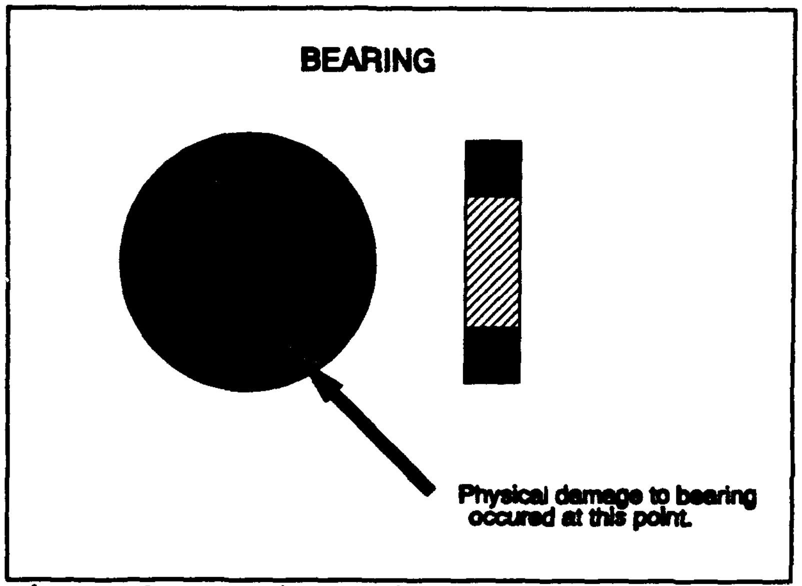


Figure 8 Bearing with damage location indicated

second segment (30 FFTs). The network was successfully trained.

The neural network responded as well to the test data as it did to the data on which it was trained. This indicates that the neural network had the ability to discern these faults and to generalize sufficiently to disregard noise. Table III contains the average values of the network's output for the test set of data

Examples of each of the 128 point FFT spectrums that were used to train the network is included in Figures 9 through 11. As can be seen, the visual detection of differences within the various frequency spectrum is often difficult.

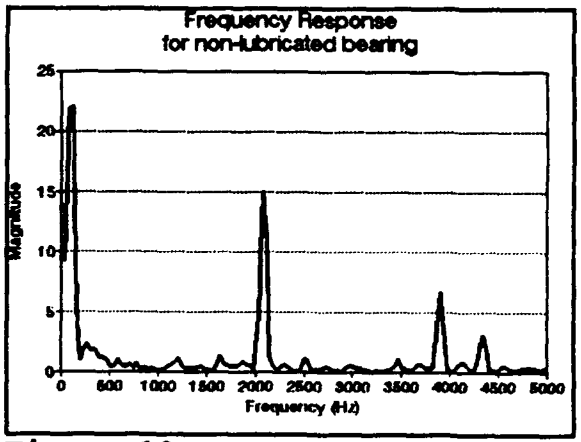


Figure 10 Frequency Spectrum of non-lubricated bearing

Disturbances

When taking measurements using an accelerometer as the sensor, the effect of noise on the system must be examined. For practical purposes, noise is defined to include sensor noise and any signal that does not originate from within the system. Previous studies have indicated that neural networks are robust and have the ability to filter out noise in the same manner that a differential amplifier reduces noise. [Ref #10]

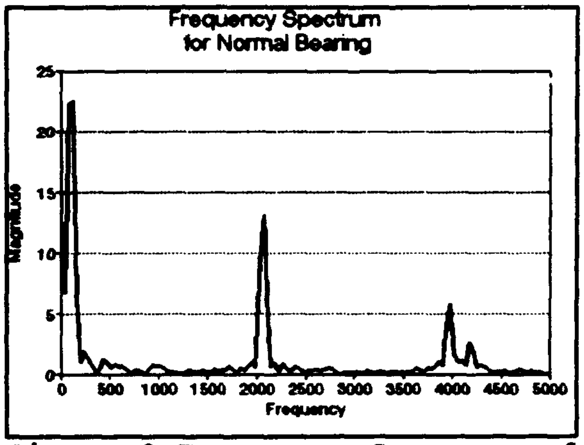


Figure 9 Frequency Spectrum of Normal Bearing

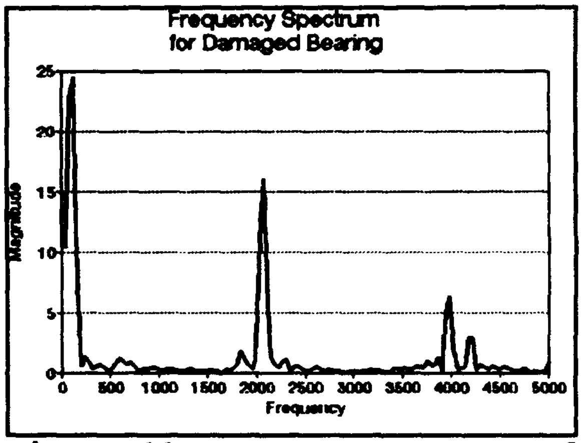


Figure 11 Frequency spectrum of damaged bearing

Table IV: Response of the output layer node's to the test data for various fault conditions. Notice the similarity to Table III.

Condition	Output Node 1	Output Node 2
Normal Condition	.170	.024
Damaged	.968	.013
Non-Lubricated	.990	.981

A differential amplifier makes use of the fact that the noise on each of its two input leads will be similar if the leads are in close proximity to one another (ie. common-mode noise). The amplifier computes the difference of the signals on the two leads which cancels out the noise. In a neural network, the noise is distributed evenly across each channel. Given that there are a substantial number of input channels, the neural network is able to filter out the noise.

The network's response to disturbances is important. If the network output is unduly influenced by disturbance noise, then the conclusions that the network reaches are not accurate. In any sort of condition monitoring system, a high level of misclassifications cannot be tolerated. Thus, it is important to ensure that disturbances do not influence the network's output.

The first approach to this problem was to determine if disturbances had a detrimental effect on the network's ability to make classifications. This was accomplished by training the network to recognize certain fault conditions and then subjecting the system to disturbances. The trained network was then tested on these new data. The result was that the output of the neural network varied with the disturbances, which indicated that the disturbances had a negative influence the network's ability to make a classification.

The next step involved determining if the network could be trained to classify data accurately in the presence of disturbance noise. This test was implemented by treating each disturbance as a normal condition and then training the neural network on it as well as normal conditions. This was successful. The neural network was able to train on these data. However, upon further study it became apparent that this experiment was too narrowly confined.

The data on which the network was tested consisted of vibration data collected at the same time that the test data were gathered. This meant that the networks ability to detect this specific disturbance did not correlate to how it would detect disturbances of a different nature. This conclusion was confirmed upon further testing of the network. In this test, data on disturbances of different types were collected. The network could be trained on specific disturbances, but did not have the ability to make generalizations on other disturbances. The problem with this system is that there is no guarantee that the network could detect disturbances upon which it was not trained.

A neural network based condition monitoring system must be able to handle all disturbance conditions that it might face. These disturbances may vary in magnitude or frequency, but the network is still required to identify them. The solution that was used in this problem was similar to a method used by Chow and Yee. Their procedure involved the use of a series of time delay mechanisms which fed the same information to the neural network at different times. [Ref #11]

The technique used in this experiment was a time averaging of the output values of the neural network. By averaging the values of the network's output over 50 iterations, the general trend of the data quickly became apparent. This can be verified visually by looking at the output of the neural network (see Figure 12). The desired output for this data was 0.9. The basis for this technique is the fact that no classification of fault/no fault for a piece of machinery should be based on a single 128-point FFT.

Care must be taken to ensure that the standard deviation of network's output is not exceptionally large. The standard deviation was not calculated in this experiment, but the outputs were perused visually to determine if a general trend did exist within the neural network's output. Figure 13 contains a sample of data where the standard deviation is large. No conclusions can be drawn on these data because of this erratic nature. However, a trend did exist in most of the data which was tested. The network output values tended to form around a certain of output as was expected (Figure 12).

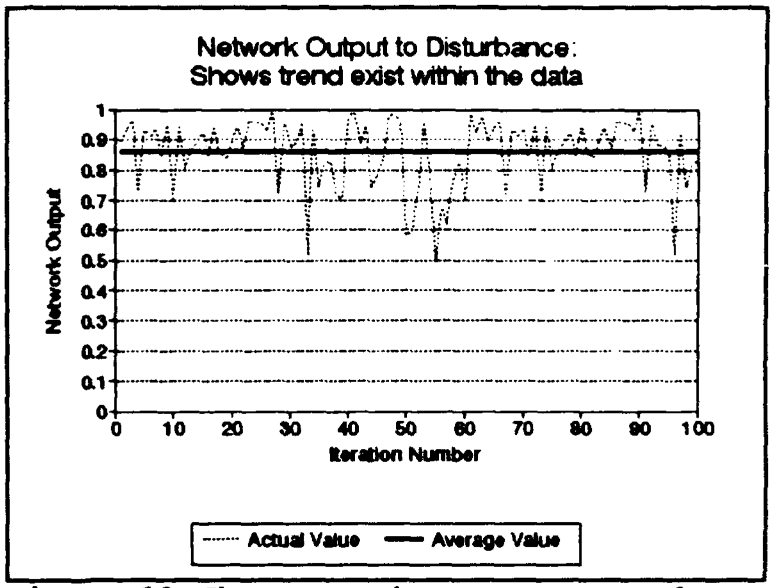


Figure 12 The network's output tends to vary around a mean of .861 with a small standard deviation.

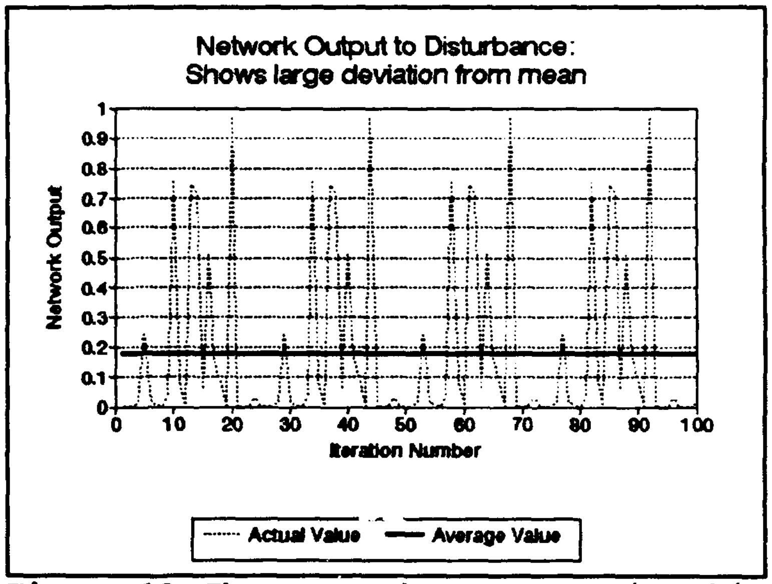


Figure 13 The network's output in this graph is volatile. The high standard deviation makes it difficult to consider these data as valid.

The time averaging technique used in this portion of the experiment acted as a filter which reduced the effects of disturbances within the network. It is important to note that this filter was not constructed to filter out disturbances which were produced by the system itself. The goal was to eliminate disturbances that were created external to the system and served to degrade the actual system data. The information that is being examined with respect to the system's vibration characteristics is periodic and should still be reflected in the frequency spectrum that is presented to the neural network.

Problems Encountered

One of the largest problems faced during the course of this research was the changing vibrational characteristics of the physical system itself. There were a lot of parts to the physical setup that could be altered and adjusted. Each time the machine was adjusted, there existed the possibility that this would in some way change the frequency spectrum of the machine. This changing spectrum was a cause of concern because of the direct influence that it played in the networks ability to detect and classify faults.

In order to eliminate this effect, all of the data for each specific fault condition were collected at one time, without changing the setup. However, in most cases when the specific condition was simulated again after a change in the setup, the neural network was not able to correctly diagnose the condition of the machinery. This network robustness problem needs to be addressed in the future.

Another potential problem that was examined dealt with the possibility that the neural network's output was a function of a few select input nodes. Such a feature of the neural network is not desirable because it indicates that the network's success is too dependent on these few input nodes. This possibility was raised due to the high level of the network's sensitivity, as evidenced by its training on disturbance noise and its response to slight variations in the frequency spectrum.

This theory was tested in this project by looking at the network's response after twenty of the network's nodes had been disabled. This experiment was repeated three times with twenty different nodes each time and the network still responded accurately to the input data. This technique was also used in a report written by Prof Clement and similar results were achieved. [Ref #12]

CONCLUSION

Based on the results of this project, several interesting conclusions can be drawn. The first is that the best way to handle disturbance noise that is generated external to the physical system being studied is to implement some sort of filtering technique. In this project, the time averaging technique was found to be satisfactory.

This experiment demonstrated that a balance must achieved in terms of the amount of pre-processing that needs to be done to the signal in order to ensure that the neural network can perform the required feature extraction. There is limited research in this area, which is why current research into neural network based condition monitoring systems relies heavily on pre-processing the data. This project attempted to show that such pre-processing was unnecessary. The results for this portion of the project were mixed. In general, the level of pre-processing that is required must be determined for each set of data.

Another conclusion that can be drawn from this project is the ability of the neural network to identify re-occurring transients. In order to recognize these transients, the neural network must break the transient into a series of steps and train on all of the steps. Essentially, the network treats each transient step as a normal operating condition and classified the information appropriately.

The final conclusion that can be drawn is the fact that the backpropagation neural network does not have the ability to make classifications on a condition on which it has not been trained. If the data were not included in the training set, then the neural network's output cannot be accurately predicted. The network's output is said to be unconstrained for these sets of data.

The results of this project indicate that there is a future for neural network based condition monitoring systems. They are valuable tools that can help the Navy with its machinery diagnostics problems. Future study should focus on constructing an operational neural network that incorporates all of the various aspects that have been addressed in this project.

Further study must also be done on determining the optimal architecture for each application of neural networks. Currently, the trial and error method is used until a satisfactory neural network is found. This is inefficient. A set of guidelines would save a lot of time and effort.

REFERENCES CITED

1.1. W.S. McCulloch and W. Pitts, "A Logical Calculus of the Ideas Imminent in Nervous Activity.", Bulletin of Mathematical Biophysics, Vol. 5, 1943, p115-133.

2. D. Hebb, The Organization of Behavior, Wiley, 1949.

3. F. Rosenblatt, "The Perceptron: A Probabilistic Model for Information Storage and Organization in the Brain", Psychological Review, Vol. 65, 1958, p386-408.

4. B. Widrow and Hoff, "Adaptive Switching Circuits", 1960 IRE WESTCON Convention Record. IRE, 1960, p96-104.

5. Marvin Minski and Seymour Pappert, Perceptrons, MIT Press, 1969.

6. D. Rummelhart and J. McClelland, Parallel Distributed Processing: Explorations in the Microstructures of Cognition Vol.1:Foundations, MIT Press, 1986.

7. Halbert White. "Learning in Artificial Neural Networks: A Statistical Computation", Neural Computation, 1989, p425-429.

8. S. Braun, Mechanical Signature Analysis: Theory and Applications, Harcourt Brace and Jovanovich, 1986, p 197-198.

9. W. D. Strunk. "Consideration for the Establishment of a Machinery Monitoring and Analysis Program for Surface Ships of the U.S. Navy", Proceedings of the 6th International Modal Conference: Vol II. Society of Experimental Mechanics, 1988, p 914-920.

10. Neural Computing: NeuralWorks Professional II/Plus. NeuralWare, Inc. 1991.

11. Mo-yuen Chow and Sui Oi Yee. "Methodology for On-Line Incipient Fault Detection in Single Phase Squirrel Cage Motors Using an Artificial Network.", IEEE Transactions on Energy Conversion. Vol 6. No3, Sep 91.

12. William Clement. "Applications of Neural Networks to Shipboard Acoustic Data Analysis". NSWC Carderock. 1993.

BIBLIOGRAPHY

Bartlett, E.B. and R. Uhrig. "Nuclear Power Plant Status Diagnostics Using an Artificial Neural Network", Nuclear Technology, vol 97, March '92, 272-281.

Berardinis, Lawrence. "Untangling the Mystery of Neural Network",Machine Design, vol 65, Jun 25, 1992.

Braun, S. Mechanical Signature Analysis: Theory and Applications, London: Harcourt Brace and Jovanovich, 1986.

Chow, Mo-yuen and Sui Oi Yee. "Methodology for On-Line Incipient Fault Detection in Single Phase Squirrel-Cage Motors Using Artificial Neural Networks", IEEE Transactions on Energy Conversion, Vol 6., No 3, Sep 91.

Clement, William. "Applications of Neural Networks to Shipboard Acoustic Data Analysis", NSWC Carderock, 1993.

Hebb, D. The Organization of Behavior, Wiley, New York, 1949.

Jackson, Leland. Signals, Systems, and Transforms, Reading, Addison-Wesley Publishing Company, Inc. 1991.

Kernighan, Brian and Dennis Ritchie. The C Programming Language, Englewood Cliffs, Prentice-Hall, 1978.

Khanna, Tarun. Foundations of Neural Networks, Reading, Addison-Wesley Publication Company, Inc. 1990.

McCulloch, W.S. and Pitts, W. "A Logical Calculus of Ideas Imminent in Nervous Activity", Bulletin of Mathematical Biophysics, Vol 5, 1943.

Minski, Marvin and Seymour Pappert. Perceptrons , MIT Press, Cambridge, 1969.

Neural Computing: NeuralWorks Professional II/Plus, NeuralWare 1991.

Pao, Yoh-Han. Adaptive Pattern Recognition, Reading, Addison-Wesley Publication Company, Inc., 1989.

Rosenblatt, F. "The Perceptron: A Probabilistic Model for Information Storage and Organization in the Brain", Psychological Review, Vol 65, 1958.

Rummelhart, D. and J. McClelland. Parallel Distributed Processing: Explorations in the MicroStructures of Cognition Vol I: Foundations, MIT Press, Cambridge, 1986.

Sorsa, Timo and Heikki Koivo. "Neural Networks in Process Fault Diagnosis, IEEE Transactions on Systems, Man and Cybernetics, Vol 21, No 4., July/August 1991.

Strunk, W.D. "Considerations for the Establishment of a Machinery Monitoring and Analysis Program for Surface Ships of the U.S. Navy", Proceedings of the 6th International Modal Analysis Conference, Vol. II, Society of Experimental Mechanics, Union College, Schenectady, New York, 1988, pp 914-920.

Urhig, Robert. "Potential Applications of Neural Networks to the Operation of Nuclear Power Plants", Nuclear Safety, Vol 32, No 1, January-March 1991.

Venkatsasubramanian, Venkat and King Chan. "A Neural Network Methodology for Process Fault Diagnosis", AIchE Journal, Vol 35, No 12, Dec 1989.

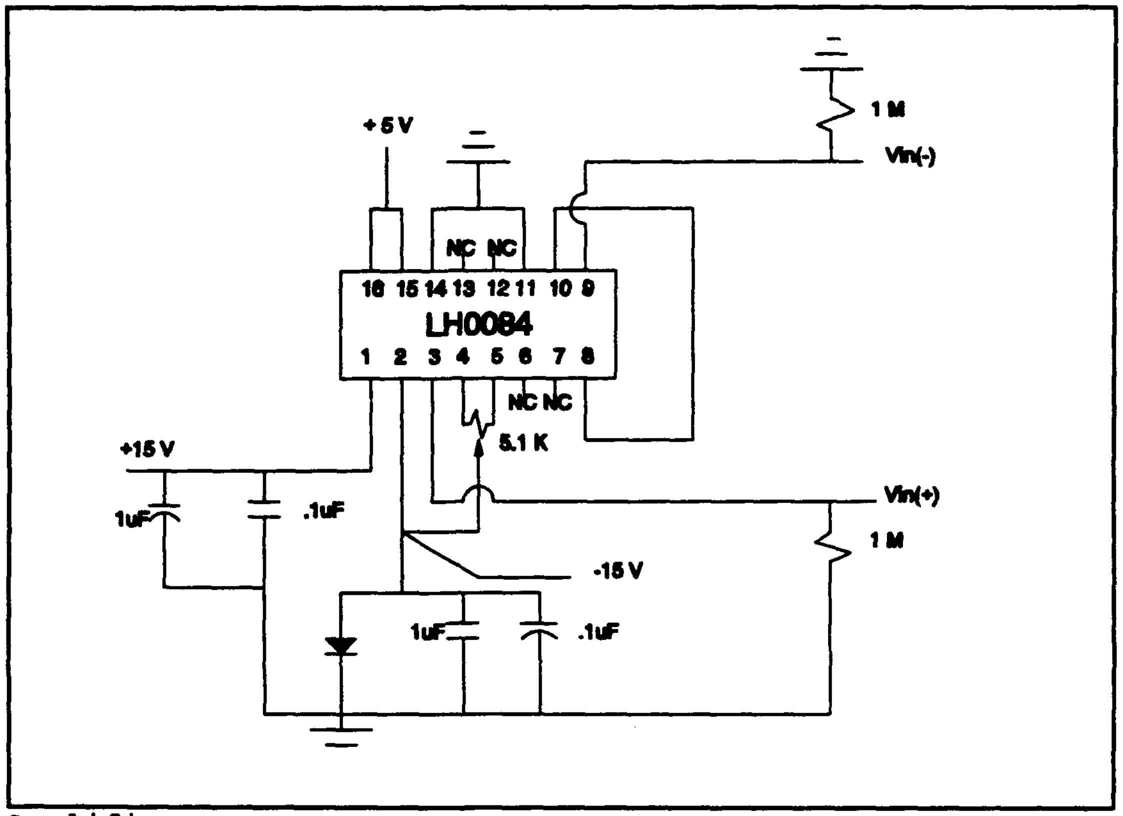
White, Halbert. "Learning in Artificial Neural Networks: A Statistical Computation", Neural Computations, vol 1, 1989, p425-426.

Widrow, B. and W. Hoff, "Adaptive Switching Circuits", 1960 IRE WESTCON Conference Record, New York, IRE, 1960.

Zurada, Jacek M. Introduction to Artificial Neural System. West Publishing Company, New York, 1992.

APPENDIX I

Amplifier Schematic



Amplifier

APPENDIX II
Data Handling Program

```
/*      "VIB1.C" -- real data from DT2801 card.  */
/* written by Assistant Professor William Clement

#include <math.h>
#include <stdio.h>
#include <stdlib.h>
#include <dos.h>
#include <process.h>
#include "pclerrs.h"  /* Include PCLAB error codes for Turbo C             */
#include "pcldefs.h"  /* Include PCLAB function declarations for Turbo C    */

typedef    struct complexx {
                              double     Re,Im;
                   };

#include "fft_inc.h"
#include "p_lib.h"

#define    POINTS                    256                         /* a power of 2 */

#define TIMING_SOURCE     0                           /* Software trigger, internal clock */
#define    START_CHANNEL 0
#define    END_CHANNEL               0
#define    GAIN                      1                           /* A/D conversion gain */

extern unsigned _stklen = 0x2000;          /* Set stack size to 8K. */

/* :-------------------------------------------------- */
main()
{
          int                   i, j, *ad_data, error=0, key_input, option, file_flag=0;
          float      clock_frequency=10000.0;          /* Block read freq. (HZ) */
          double     *time_axis, *freq_axis, *data;
          double     net_out=0;
          char       fname[20];
          FILE       *fp;

          if ( (ad_data=(int *) calloc(POINTS, sizeof(int)))==NULL )
                     exit(1);
          if ( (time_axis=(double *) calloc (POINTS, sizeof(double)))==NULL )
                     exit(1);
          if ( (freq_axis=(double *) calloc (POINTS, sizeof(double)))==NULL )
                     exit(1);
          if ( (data=(double *) calloc (POINTS, sizeof(double)))==NULL )
                     exit(1);
          for (i=0; i<POINTS; i++) {
                     time_axis[i] = i/clock_frequency;
                     freq_axis[i] = i * clock_frequency/POINTS;
          }
          printf("Save an output data file? (y/n)  ");
          i = getch();
          if (i=='y' || i=='Y') {
                     printf("Name of OUTPUT FILE:  ");
                     scanf("%s",fname);
                     fp = fopen(fname,"w");
                     file_flag=1;
                     printf("Desired Network Output:  ");
                     scanf(" %lf",&net_out);
```

```
}

/* Initialize the PCLAB routines and address board #1:  */
initialize();
select_board(1);
error = setup_adc(TIMING_SOURCE, START_CHANNEL, END_CHANNEL, GAIN);
error = set_clock_frequency(&clock_frequency);

do {
        system("cls");
        printf("Your options are:\n");
        printf("  (0)  QUIT, or\n");
        printf("  (1)  Plot TIME-DOMAIN data, or\n");
        printf("  (2)  Plot FREQUENCY-DOMAIN data WITH Hanning Window.\n");
        printf("  (3)  Plot FREQUFNCY-DOMAIN data WITHOUT Hanning Window.\n");
        printf("Option:  ");
        scanf(" %d",&option);
} while (option<0 || option>3);

while (option!=0) {
        /* Perform a series A/D conversion:  */
        disable_system_clock();
        error = adc_series (POINTS, ad_data);
        enable_system_clock();

        for (i=0; i<POINTS; i++)
                data[i] = (ad_data[i]-2048)/204.8;
        switch (option) {
                case 1:
                        plot_data (time_axis, data, POINTS, 2);
                        if (file_flag) {
                                for (i=0, j=0; i<POINTS; i++, j=(++j)%5) {
                                        fprintf(fp,"%.5e ",data[i]);
                                        if (j==0 && i!=0 && i<POINTS-1)
                                                fprintf(fp,"\n& ");
                                }
                                fprintf(fp," %.5lf\n",net_out);
                        }
                        break;
                case 2:
                        Hanning_real (data, POINTS);
                case 3:
                        FFT_magnitude (data, POINTS);
                        plot_data (freq_axis, data, POINTS/2, 2);
                        if (file_flag) {
                                for (i=0, j=0; i<POINTS/2; i++, j=(++j)%5) {
                                        fprintf(fp,"%.5e ",data[i]);
                                        if (j==0 && i!=0 && i<POINTS/2-1)
                                                fprintf(fp,"\n& ");
                                }
                                fprintf(fp," %.5lf\n",net_out);
                        }
                        break;
                default:
                        break;
        }

        if (kbhit()) {
                getch();
                while (kbhit())
                        getch();
                set_mode("text");
```

```
                        do {
                                system("cls");
                                printf("Your options are:\n");
                                printf("  (0)  QUIT, or\n");
                                printf("  (1)  Plot TIME-DOMAIN data, or\n");
                                printf("  (2)  Plot FREQUENCY-DOMAIN data WITH Hanning Window.\n");
                                printf("  (3)  Plot FREQUENCY-DOMAIN data WITHOUT Hanning Window.\n");
                                printf("Option:  ");
                                scanf(" %d",&option);
                        } while (option<0 || option>3);
                }
                else
                        delay (10);
        }

        if (file_flag)
                fclose(fp);
        terminate();
}
/* ---------------------------------------------------------------- */
```